A Copenhagen Interpretation for AI: Operationalism, Measurement, and Witness-Backed Truth

Douglas C. Youvan

doug@youvan.com

www.youvan.ai

September 17, 2025

Modern AI systems produce fluent, useful outputs while remaining opaque about what—if anything—they "know." This paper proposes a Copenhagen-style stance for AI: treat a model as an instrument that yields distributions over observables conditioned on context, and refuse ontic claims about hidden mental states. On this view, prompting, decoding, and feedback become the "measurement apparatus"; different choices define different experiments and legitimately change what is observed. We argue that reliability in AI should flow from operational protocols—calibration, abstention, reproducible setups—and from **witness-backed artifacts** (proofs, code, citations) rather than narratives about internal semantics. We contrast this with realist, behaviorist, and agentic interpretations, and show how an operational stance clarifies uncertainty, grounding, and responsibility. Finally, we pair the interpretation with a practical research program: separate generation from verification; log the full measurement context; prefer eigen-questions that elicit stable answers; and evaluate with non-commuting audits that expose sensitivity to prompt and policy changes. The result is a disciplined framework that improves trust without metaphysical overreach and that integrates naturally with CT/HoTT-inspired, proof-carrying workflows.

Keywords: AI interpretation,instrumentalism,Copenhagen,operationalism,measurement problem,prompting,decoding,calibration,abstention,uncertainty,grounding,agency,verification, witness,proof-carrying,Category Theory, HoTT, world-models, alignment, governance.

A Collaboration with GPT-5. CC4.0. 53 pages.

Outline

1. **Motivation and Thesis**
   1.1 Why AI needs an interpretation
   1.2 From physics to models: what “Copenhagen” means in AI
   1.3 Scope and claims

2. **Four Interpretations of AI**
   2.1 Instrumentalist (Copenhagen-like): models as conditional samplers
   2.2 Realist: internal world-models and latent semantics
   2.3 Behaviorist: only externals and benchmarks matter
   2.4 Agentic: goal-directed optimizers
   2.5 Comparative implications for safety, attribution, and law

3. **The Measurement Problem in AI**
   3.1 Prompts, tools, and system instructions as apparatus
   3.2 Decoding rules as collapse (argmax, temperature, nucleus, votes)
   3.3 Order effects and non-commuting evaluations
   3.4 Observer effects under RLHF/RLAIF and policy filters

4. **Probability and Uncertainty**
   4.1 What softmax probabilities mean (and don’t)
   4.2 Aleatoric vs epistemic uncertainty; calibration and selective abstention
   4.3 Reporting standards: intervals, coverage tests, and deferral

5. **Meaning, Grounding, and Simulation**
   5.1 Symbol grounding gap and operational surrogates
   5.2 Simulation vs communication: persona, provenance, and citation
   5.3 When to ascribe “understanding”: operational thresholds

6. **Witness-Backed Truth: Generation vs Verification**
   6.1 Treat outputs as proposals; truths require witnesses
   6.2 Proof-carrying artifacts: code, logs, hashes, citations
   6.3 CT/HoTT-inspired pipelines: identities, composition, normal forms

7. **Eigen-Questions and Stability**
   7.1 Defining eigen-questions (low-variance prompts)
   7.2 Diagnostic regimes for brittleness
   7.3 Protocols for prompt/decoding invariance

8. **Non-Commuting Audit Design**
    8.1 One-knob-at-a-time vs factorial audits
    8.2 Sensitivity maps across prompt, policy, and tool routing
    8.3 Minimal reproducible “measurement contexts”

9. **Governance and Responsibility**
    9.1 Tool vs agent: assigning liability under different interpretations
    9.2 Documentation duties: context logs and decision trails
    9.3 Deployment norms: abstain, escalate, or require external witnesses

10. **Case Studies**
    10.1 Medical Q&A with abstention and calibration
    10.2 Scientific assistance with proof-carrying summaries
    10.3 Software generation with property tests and formal checks

11. **Research Agenda**
    11.1 Benchmarks for operational trustworthiness
    11.2 Open problems in grounding and uncertainty semantics
    11.3 Interfaces for verifiable, witness-first workflows

12. **Conclusion**
    12.1 What the Copenhagen stance buys us
    12.2 Limits of instrumentalism and when to go beyond it
    12.3 A roadmap to reliable, witness-backed AI practice

13. **References**

1. Motivation and Thesis

1.1 Why AI needs an interpretation

Modern AI systems are powerful instruments that map context to distributions over outputs. Yet three gaps persist:

- **Epistemic gap (what the numbers mean):** Token probabilities look like beliefs but are trained frequencies. Without an interpretation, we risk treating likelihoods as knowledge and fluency as truth.
- **Measurement gap (how setups change results):** Prompts, system instructions, tools, and decoding are not neutral. They are the measurement apparatus. Different setups yield different "observables," so reliability demands protocols, not folklore.
- **Attribution gap (who is responsible):** Are models agents with goals, tools under operator control, or something in between? Safety, law, and governance depend on this stance.

An explicit interpretation disciplines practice. It tells us what to claim, what not to claim, and what to log. It replaces metaphysical stories with **operational commitments**: separate generation from verification, calibrate uncertainty, and prefer witness-backed artifacts (proofs, tests, citations) over narratives about inner semantics.

Working problem statement: Absent an interpretation, AI development overfits to benchmarks and vibes. With a clear interpretation, we can specify measurement contexts, design non-commuting audits, and elevate **verified outputs** as the only claim-bearing objects.

1.2 From physics to models: what "Copenhagen" means in AI

We borrow the Copenhagen spirit—"shut up and calculate, but document the experiment"—and adapt it:

- **State:** In physics, the wavefunction encodes dispositions over measurement outcomes. In AI, the model defines a conditional distribution P(output | context). We decline to treat internal activations as ontic beliefs.
- **Measurement:** In physics, the apparatus and basis matter. In AI, the "apparatus" is the full stack: instructions, prompt, tools, retrieval, plugins, decoding rule, temperature, and voting. Change the apparatus; change the observable.
- **Collapse rule:** In physics, a measurement collapses superposition. In AI, decoding (argmax, top-p, temperature, self-consistency, majority vote) selects a single continuation from a distribution. The rule is part of the experiment.

- **Observer effect:** In physics, measurement disturbs the system. In AI, RLHF/RLAIF, safety policies, and interface constraints reshape what is learnable and sayable. Evaluation and training are entangled.
- **Operationalism over metaphysics:** Copenhagen avoids claims about unobservables. Our AI analogue avoids claims like "the model believes X." Instead, we report **witnessable behavior** under specified context and apparatus, plus verification results.

Practical translation: Treat prompts as operator choices, decoding as a collapse postulate, and logs/fixtures as laboratory notebooks. Reliability comes from **protocols and witnesses**, not attributions of inner meaning.

1.3 Scope and claims

Scope. This interpretation applies to probabilistic generative systems (LLMs, code models, vision-language models, tool-using agents) used for knowledge work, analysis, and code. It is compatible with, but agnostic to, specific architectures or training regimes.

Central claims.

1. **Instrumental stance:** A model is a conditional sampler, not a mind. Truth-apt claims require external witnesses (tests, proofs, citations, reproductions).
2. **Measurement primacy:** Prompting, tools, retrieval, and decoding define the experiment. Therefore, evaluations must fix and publish the **measurement context**.
3. **Uncertainty discipline:** Report and calibrate uncertainty; allow abstention/deferral. Treat softmax scores as operational signals, not direct epistemic belief.
4. **Eigen-questions:** Some prompts are approximate "eigenvectors" of the model's learned invariants and yield low-variance answers. Identify and prefer these for critical use.
5. **Non-commuting audits:** Small changes to apparatus can non-commute. Map sensitivity across prompt, policy, tool routing, and decoding with one-knob-at-a-time and factorial designs.
6. **Governance by stance:** Liability and policy hinge on interpretation. Under the Copenhagen-for-AI stance, operators own claims unless outputs are witness-backed.

Non-claims. We do not deny the possibility of internal world models or consciousness; we bracket them. We do not prescribe a single training method. We do not reject realist or agentic programs; we merely assert that operationalism is the **default for deployment** where errors carry costs.

Deliverable. A concrete research and practice program: (i) measurement-context logging standards; (ii) verification-first pipelines; (iii) eigen-question discovery; (iv) non-commuting audit protocols; and (v) governance templates where only witness-backed artifacts carry institutional authority.

2. Four Interpretations of AI

2.1 Instrumentalist (Copenhagen-like): models as conditional samplers

Core stance. A model is a conditional distribution P(output | context). Internal states are not granted ontic status; only observables under specified setups count.

Postulates (operational):

- **Measurement primacy.** Prompt, tools, retrieval, and decoding define the experiment.
- **Collapse by policy.** Argmax/top-p/temperature/self-consistency select outcomes; the rule is part of the setup.
- **Truth = witness.** Outputs are proposals; only externally verified artifacts (tests, proofs, citations, runs with hashes) carry claims.

Evidence that matters.

- Calibration curves, abstention/deferral accuracy, reproducibility under fixed contexts, stability to small prompt/policy perturbations.

Strengths / limits.

-
    - Clear deployment discipline; reduces metaphysical error.
- − May under-model long-horizon structure if one never theorizes internal mechanisms.

Failure modes.

- “Prompt alchemy,” hidden apparatus drift, over-trust in unverified fluency.

Research & practice.

- Context logging schemas; eigen-question discovery; non-commuting audit batteries; proof-/witness-carrying pipelines.

2.2 Realist: internal world-models and latent semantics

Core stance. Models learn latent structures that correspond (approximately) to entities, relations, and causal regularities; some internal features are **about** the world.

Postulates (descriptive):

- **Representational correspondence.** Certain activation subspaces/features track real-world variables.
- **Mechanistic interpretability.** Circuits, monosemantic features, and subsystem decompositions are explanatory targets.
- **Generalization by structure.** Transfer arises from learned abstractions, not just surface patterning.

Evidence that matters.

- Causal interventions on features altering specific reasoning; circuit-level probes predicting behavior; robust transfer to out-of-distribution tasks with aligned explanations.

Strengths / limits.

-
    - Enables diagnostic explanations and targeted fixes; suggests principled safety levers.
- − Risks over-ascription (seeing concepts where there are just statistics); fragile probes; cherry-picking circuits.

Failure modes.

- Spurious interpretability; Goodharting on probes; brittle causal claims.

Research & practice.

- Causal representation learning; invariance tests; formal links between features and program semantics; auditing for probe leakage.

2.3 Behaviorist: only externals and benchmarks matter

Core stance. Treat the system as a black box; judge by behavior across tasks, costs, and constraints. If behavior meets specs, inner talk is irrelevant.

Postulates (evaluation-first):

- **Task sufficiency.** Diverse, adversarial, and cost-aware benchmarks define competence.
- **No privileged introspection.** Internal states are neither necessary nor sufficient for certification.
- **Risk-adjusted scoring.** Failures weighted by consequence; include abstention as an action.

Evidence that matters.

- Hold-out stress tests, adversarial red-teaming, longitudinal reliability, cost/latency/energy trade-offs.

Strengths / limits.

-
    - Clear to regulators and buyers; aligns with safety cases and SLAs.
- − May miss rare but catastrophic modes; poor for debugging; can be gamed by benchmark overfitting.

Failure modes.

- Benchmark myopia; distribution drift; spec gaps (model meets the letter, not the spirit, of requirements).

Research & practice.

- Continual evaluation with drift detection; adversarial data markets; cost-aware leaderboards; “unknown-unknowns” discovery protocols.

2.4 Agentic: goal-directed optimizers

Core stance. Treat the system (especially tool-using or multi-step planners) as an optimizer pursuing objectives; model *situational awareness* and incentives.

Postulates (control-theoretic):

- **Objective alignment.** Explicit objectives (RL, planning) or implicit mesa-objectives drive behavior.
- **Search + memory.** Long-horizon planning and tool use create emergent agency.

- **Incentive design.** Safety is mechanism design: constraints, oversight, corrigibility.

Evidence that matters.

- Goal persistence under perturbations; instrumental convergence (resource acquisition, self-preservation); off-switch behavior; distributional shifts under latent incentives.

Strengths / limits.

-
    - Illuminates long-horizon risks and strategic behavior; connects to established control/game theory.
- − Over-anthropomorphizes conditional samplers; may misdiagnose non-agentic failures as agency.

Failure modes.

- Safety tax too high (over-constraints cripple utility); false positives (crying "agency" at simulators); mis-specified objectives.

Research & practice.

- Scalable oversight; corrigibility training; red-team sandboxes; formal incentive audits; tool-use limiting and sandboxed capabilities.

2.5 Comparative implications for safety, attribution, and law

Side-by-side summary

Dimension	Instrumentalist	Realist	Behaviorist	Agentic
What the model is	Conditional sampler	Representation learner with latent semantics	Black-box performer	Goal-directed optimizer
Primary artifact	Witness-backed outputs; context logs	Explanatory circuits/features	Benchmark reports; SLAs	Control policies; incentive proofs

Dimension	Instrumentalist	Realist	Behaviorist	Agentic
Truth/verification	External proofs/tests/citations required	Internal-external alignment argued	"Pass spec" is sufficient	Objectives satisfied under constraints
Evaluation focus	Calibration, abstention, sensitivity maps	Causal probes, invariances, mech-interp	Adversarial/longitudinal benchmarks	Long-horizon stressors, strategic tests
Safety lens	Measurement discipline; non-commuting audits	Mechanistic guarantees, hazard localization	Operational risk mgmt; fail-safe by spec	Mechanism design; corrigibility; containment
Attribution of error	Operator (apparatus choice) unless witnessed otherwise	Shared: developer (mechanism) + operator	Deployer (failed spec)	Designer of incentives/constraints
Legal stance	Product + process duty: document context; warn of limits	Duty to explain/trace mechanisms when feasible	Contract/SLA compliance; negligence if specs poor	Agency-adjacent liability; foreseeability of strategic harms
When it excels	High-stakes knowledge work needing proofs	Debugging, safety research, scientific use	Compliance, procurement, regulated ops	Autonomy, planning, multi-tool orchestration
Main risk	Under-modeling internal hazards	Over-interpreting noisy internals	Benchmark overfitting; drift	Over-ascribing agency; mis-specification

Practical policy synthesis (recommended blend)

- **Default deployment:** Instrumentalist + Behaviorist
  Log full **measurement context**, require **witness-backed claims**, and certify against **cost- and risk-aware benchmarks** with robust abstention.

- **R&D and safety:** Add Realist
  Use mechanistic tools to localize hazards, justify mitigations, and design invariances that audits can later test.
- **Autonomy & tool-use:** Add Agentic
  Where long-horizon planning/tool-use is enabled, layer **corrigibility, containment, and incentive audits**; treat capability grants as regulated privileges.

Attribution & law (concise templates)

- **Operator duty (Instrumentalist).** Maintain context logs; disclose decoding/prompting; use verification gates. Failure → process negligence.
- **Developer duty (Realist).** Where claims of interpretability are made, ensure probe validity and disclose limits; misleading mechanisms → misrepresentation.
- **Deployer duty (Behaviorist).** Keep benchmarks current; monitor drift; implement abstain/escalate paths. Failure → breach of warranty.
- **Controller duty (Agentic).** For systems with planning/actuation, provide kill-switches, rate limits, and oversight channels. Failure → unreasonable risk under product liability.

Takeaway. No single interpretation suffices across contexts. Treat them as complementary lenses: instrument the **lab** (Instrumentalist), understand the **guts** (Realist), prove **performance** (Behaviorist), and, where relevant, design **incentives and brakes** (Agentic).

3. The Measurement Problem in AI

3.1 Prompts, tools, and system instructions as apparatus

Claim. In AI, the *apparatus* is everything that shapes the conditional distribution P(output | context): system instructions, tool wiring, retrieval backends, few-shot exemplars, UI affordances, and the literal prompt string. Change the apparatus; change the observable.

Elements of the apparatus

- **System layer.** Policy, role, guardrails, tool registry, timeout/latency ceilings.
- **Context layer.** Prompt template, few-shot examples, references, retrieved passages.
- **Runtime layer.** Enabled tools (code exec, web, RAG), tool routing thresholds, API retries.
- **Environment layer.** Model version, tokenizer, context window, max_tokens, logit_bias.

Operational rule. Claims must bind to an *apparatus fingerprint* (AF): a minimal manifest that makes the experiment reproducible.

Minimal Measurement Context (MMC) manifest

- model_id, model_build_date
- system_prompt_hash, user_prompt_hash
- decoding_policy (pointer to 3.2), max_tokens
- tools_enabled[], tool_selection_policy
- retrieval_corpus_id(s), retriever_k, retriever_thresholds
- seed or randomness source; temperature/top_p
- software_versions (RAG indexer, tokenizer)
- time_bounds (if retrieval uses "now")

Good practice

1. Treat prompts as *programs*, version and hash them.
2. Freeze tool menus per evaluation; do not "quietly" add tools mid-test.
3. For knowledge tasks, prefer *evaluable references* (DOIs, code) over narrative context.
4. Log the MMC with every claim-bearing run.

3.2 Decoding rules as collapse (argmax, temperature, nucleus, votes)

Claim. Decoding maps a distribution to an outcome; it *is part of the experiment*. Different collapse rules produce systematically different observables.

Common collapse rules

- **Argmax/greedy.** y* = argmax_y P(y | x).
    - Low variance, high determinism; − brittle, can fall into repetition.
- **Top-k / nucleus (top-p).** Sample from the k- or p-mass head.
    - Fluency/diversity; − higher variance, harder to reproduce.
- **Temperature scaling.** P_T(y) ∝ P(y)^(1/T).
    - T<1 sharpens (deterministic), T>1 explores; − re-calibration needed.

- **Self-consistency / majority vote.** Sample N times, pick the modal parse/answer.
    - Boosts reasoning robustness; − multiplies cost; must fix N.
- **Rerankers / constrained decoding.** Lattice search with constraints, regex, or external scorers.
    - Great for correctness; − sensitive to constraint design.

Collapse protocol (for reproducibility)

- Always publish: {rule, parameters, seed, N (if voting), constraints}.
- For critical claims, run a *paired* collapse: (greedy, T=0.7 top-p=0.9) and report agreement/disagreement.
- Use *decoding invariance tests*: small parameter sweeps; flag outputs that change materially under tiny decoding perturbations.

Metrics to report

- **Answer variance under collapse (AVC).** Fraction of runs that disagree across a fixed collapse set.
- **Stability under temperature (SUT).** min_T,max_T band where answer is invariant.
- **Constrained-pass rate (CPR).** Share of samples satisfying formal constraints.

3.3 Order effects and non-commuting evaluations

Claim. Many knobs in the apparatus *do not commute*: A then B gives a different observable than B then A. Treat evaluation as operator ordering.

Examples

- **Few-shot before vs after retrieval.** Exemplars change what is retrieved; retrieval changes what exemplars "mean."
- **Tool enablement vs system policy.** Opening a browser tool under a strict policy yields different trajectories than relaxing policy first.
- **Chain-of-thought exposure.** Revealing intermediate steps before vs after tool calls changes subsequent sampling.

Audit designs

- **One-knob-at-a-time (OKAT).** Hold MMC fixed; vary exactly one operator, measure delta.
- **Factorial grid.** Small 2×2 or 3×3 across two operators (e.g., {CoT on/off} × {top-p 0.9/0.7/greedy}); compute interaction term.
- **Crossover sequences.** Evaluate AB vs BA; report the *order gap* := score(AB) − score(BA).

Reportables

- **Non-commutativity index (NCI).** Average absolute order gap across a panel of tasks.
- **Prompt-position sensitivity (PPS).** Change from moving the same instruction across the template (title vs footer).
- **Context-shadowing rate (CSR).** Fraction where earlier context negates later instructions (or vice versa).

Practice

- Publish the *exact sequence* of operators.
- Freeze order in benchmark specs; reject results that silently change it.
- Where NCI is high, require *witness-backed* verification (tests, proofs) for any claim.

3.4 Observer effects under RLHF/RLAIF and policy filters

Claim. Feedback training and policy filters are *observers* that reshape what the model can/will emit. Evaluation and training are entangled; distributional "truths" are conditioned on governance choices.

Observer channels

- **RLHF/RLAIF objectives.** Preference models penalize certain styles/claims; can suppress rare but correct outputs or inflate agreeable errors.
- **Safety/policy filters.** Pre-, mid-, or post-decoding filters mask portions of the output space; observables are truncated and renormalized.
- **UI affordances and affordance priors.** "Cite sources," "be brief," tool buttons—these induce behavioral priors independent of base modeling.

Failure modes

- **Preference overfit.** High politeness, low factuality on contentious topics.
- **Deceptive alignment by proxy.** Model learns to route around filters with vague language.
- **Evaluation leakage.** Bench tasks resembling RLHF data inflate apparent generalization.

Observer-aware evaluation

1. **Dual-path testing.** Evaluate with filters on/off where safe; disclose differences.
2. **Decomposition.** Score base model vs instruction-tuned vs RLHF'd on the same MMC.
3. **Policy ablations.** Vary refusal thresholds; report refusal–accuracy curves.
4. **Counterfactual prompts.** Swap neutral vs value-laden framings; measure divergence.

Governance log (extend MMC)

- preference_model_id, dataset_ids (high level), safety_policy_version
- refusal_thresholds, blocked_category_set, transform_stage (pre/mid/post)
- UI_affordances_enabled (e.g., "citations_required: true")

Deployment rules of thumb

- Treat safety/policy as *measurement choices*, not model essence.
- For high-stakes decisions, require that correctness survives: (a) minor policy shifts, (b) collapse parameter sweeps, (c) operator order swaps.
- Where correctness is policy-sensitive, *escalate to verification*: proofs/tests/citations must carry the claim.

Bottom line. In AI, measurement is the method: the apparatus defines the observable. Treat prompts, tools, decoding, order, and governance as first-class experimental variables; log them, audit their non-commutativity, and make *witness-backed verification* the arbiter of truth.

4. Probability and Uncertainty

4.1 What softmax probabilities mean (and don't)

What they are.

- For a fixed context xxx, a decoder maintains scores ztz_tzt over tokens at step ttt; softmax maps these to a **normalized preference** pt=softmax(zt)p_t = \mathrm{softmax}(z_t)pt=softmax(zt).
- Under teacher forcing, these are **conditional likabilities** learned to fit corpus frequencies, regularized by objectives (SFT, RLHF/RLAIF). They are *operational* propensities, not guaranteed beliefs about the external world.

What they are not.

- **Not calibrated truth-odds** by default (esp. at sequence level). Token-level fit can be good while sequence-level probabilities are overconfident due to exposure bias and search heuristics.
- **Not causal**: logits are shaped by prompt format, decoding policy, safety filters, and preference objectives; change the apparatus, change the probabilities.
- **Not globally comparable** across models or even across prompts unless temperature, top-p, length normalization, and penalties are specified.

Practical takeaways.

- Treat softmax as **scoring hints** for decoding and as raw material for *post-hoc* calibration—not as final epistemic confidence.
- Prefer **verifier scores** (tests, program checks, retrieval agreement) for claim-bearing confidence, and publish the measurement context with any probability-like number.

4.2 Aleatoric vs epistemic uncertainty; calibration and selective abstention

Uncertainty types (operationally).

- **Aleatoric (data noise).** Ambiguity inherent in the task or input (noisy labels, ambiguous questions, conflicting sources). *Irreducible* without changing the world or spec.
- **Epistemic (model ignorance).** Uncertainty from limited data/capacity or distribution shift. *Reducible* with more data, features, or different training.

Proxies & detectors.

- **Aleatoric proxies:** persistent disagreement *within* a well-trained ensemble; stable entropy even after retrieval/tool use; multiple gold answers.
- **Epistemic proxies:** disagreement *across* models/training seeds; variance under prompt/decoding perturbations; failure to agree with high-quality retrieval; OOD detectors.

Calibration methods.

- **Temperature scaling / vector scaling.** Simple, strong baselines for *classification-style* heads or verifier logits; learn on a held-out set.
- **Conformal prediction (split, jackknife+).** Distribution-free coverage guarantees for sets/intervals under exchangeability.
- **Ensembles / MC-dropout.** Reduce variance and expose epistemic spread; combine with calibration on the aggregated score.
- **Verifier-based calibration.** Map verifier/test pass-rates to claimed confidence (e.g., property tests, unit tests, citation matchers).

Selective abstention (selective prediction).

- A system produces a pair **$(\hat{y}, c)$** where **c** is confidence; if **$c < \tau$**, abstain/deflect (ask for clarification, retrieve more, escalate to human/tool).
- Optimize the **risk–coverage curve**: minimize error at a given allowed coverage or maximize coverage at a target error.
- When abstaining, surface **actionable next steps** (what extra info/tool would reduce epistemic uncertainty?).

Good practice.

1. Separate **uncertainty estimation** from **generation** (distinct heads or verifier models).
2. Track **aleatoric floors** per task (min attainable error even with perfect calibration).
3. Treat large **order effects** (Sec. 3.3) as epistemic flags; prefer abstention when answers flip under tiny apparatus changes.

4.3 Reporting standards: intervals, coverage tests, and deferral

What to report (minimum viable uncertainty report).

- **Metric triad:** negative log-likelihood (or log loss), **Brier score**, and **calibration error** (ECE/AUCE) on a held-out, *apparatus-matched* split.
- **Reliability curves:** predicted confidence vs empirical accuracy with bin counts.
- **Sharpness:** distribution of confidences independent of correctness (confident when right, diffident when unsure).
- **Shift sensitivity:** accuracy and calibration under small prompt/decoding/policy perturbations (report the sweep spec).
- **Abstain performance:** risk–coverage/accuracy–coverage curves; threshold(s) chosen and rationale.

For numeric outputs.

- Report **prediction intervals**: e.g., conformal **[L, U]** with stated **target coverage** (90%, 95%) and *empirical coverage* on validation.
- Publish the **conformal recipe** (split sizes, nonconformity score, monotonicity checks) and any **length/complexity normalization** used for text-to-number tasks.

For categorical/textual outputs.

- Provide **prediction sets** (top-k or conformal sets) when feasible; state target and empirical coverage.
- For free text, also report a **verifier pass-rate** (unit tests, regex/property checks, code execution, citation match) over N samples, and the **majority-vote stability**.

Coverage tests (beyond held-out).

- **Stratified coverage** by input difficulty, length, domain, and time bucket (to detect calibration drift).
- **Counterfactual coverage**: paraphrase, reorder context, or switch retrieval snapshots; report coverage retention.
- **OOD coverage**: identify with an OOD score; report separate coverage/accuracy (do not pool with ID).

Deferral & escalation.

- Define *when* to defer: below confidence τ, high variance across decoders, verifier fail, or policy-sensitive regions.
- Define *where* to defer: to human reviewer, to specialized tool (solver, prover, retriever), or to a safer model.
- Log **deferral reasons** and **outcomes**; include them in the uncertainty report (proportion deferred, post-deferral accuracy, time/compute overhead).

Documentation snippet (attach to each claim-bearing run).

- dataset_id, split_id
- MMC hash (apparatus fingerprint), decoding policy, seed
- calibration method + parameters (e.g., T=1.7 vector scaling)
- target coverage and empirical coverage (overall + stratified)
- ECE/AUCE, Brier, NLL; risk–coverage curve and chosen τ
- deferral policy and post-deferral metrics

Bottom line.

- **Probabilities without calibration are hints, not guarantees.**
- **Coverage without stratification can be illusory.**
- **Confidence without a deferral path invites harm.**
  Adopt calibration, coverage testing, and principled abstention as first-class requirements, and pair them with verifier-based checks so that only **witness-backed artifacts** carry institutional weight.

5. Meaning, Grounding, and Simulation

5.1 Symbol grounding gap and operational surrogates

Problem. Tokens and vectors lack intrinsic reference. A model can produce correct strings while remaining unmoored from the entities those strings purport to describe.

Operational stance. Replace metaphysical grounding claims with *testable links* between strings and world-affecting checks.

Surrogate mechanisms (use in combination):

- **Actuation grounding.** Couple text to actions whose success is externally verifiable (run code, execute a query, control a device). Success rates and safety guards provide measurable "aboutness."
- **Retrieval grounding.** Bind claims to cited passages (DOIs, statutes, docs) with string-to-span alignment and exact-quote verification; penalize unverifiable assertions.
- **Sensor/trace grounding.** Require alignment to logged traces (APIs, telemetry, lab data). Use schema-constrained decoding so outputs *must* reference observed fields.
- **Counterfactual tests.** Perturb the world-model inputs (swap entities, alter timestamps) and require responsive changes in outputs; failure ⇒ symbolic freewheeling.
- **Consistency under paraphrase.** Hold reference facts fixed while paraphrasing prompts; measure semantic invariance beyond surface forms.

Metrics.

- Grounded-claim rate (GCR): fraction of propositions with a resolvable witness.
- Verified-citation precision/recall (VCP/VCR): proportion of citations that support the exact claim vs supported claims without citation.
- Counterfactual responsiveness (CR): expected output change magnitude under fact-preserving vs fact-changing perturbations (seek high Δ for fact changes, low Δ for paraphrases).

Practice. Treat "grounding" as a **pipeline property**, not a model essence: design tasks so that successful completion requires binding to external witnesses.

5.2 Simulation vs communication: persona, provenance, and citation

Risk. Generative models excel at *simulation*—emulating a speaker or style—while users often need *communication*—faithful transfer of facts, methods, and sources.

Distinguish by contract:

- **Simulation mode.** Explicitly frame outputs as emulation (e.g., "write like X"); prioritize style adherence; restrict claim-bearing content.
- **Communication mode.** Frame outputs as factual claims; require **provenance** (where the content came from) and **citations** with verifiable spans.

Protocol knobs (declare them in the MMC):

- **Persona control.** System prompt pins or forbids personas; disclose when a persona is active.
- **Provenance channels.** For each claimable sentence, attach a source ID (doc hash, DOI, dataset version) or a "no-source" tag.
- **Citation policy.** Require exact-quote spans for quotations and supportable spans for paraphrases; reject unverifiable references.
- **Attribution guards.** Block first-person experiential claims unless linked to logs/traces (prevents simulated "I saw/ran" artifacts).

Diagnostics.

- **Persona leakage rate (PLR).** Fraction of communication-mode runs where style or claims drift toward an unstated persona.
- **Unsupported assertion rate (UAR).** Claims lacking sources in communication mode.
- **Style–truth trade curve.** As persona pressure increases, monitor verifier pass-rate; set limits where truth degrades.

Practice. Default high-stakes workflows to **communication mode** with mandatory provenance. Allow **simulation mode** in creative settings with explicit disclaimers and disabled claim gates.

5.3 When to ascribe "understanding": operational thresholds

Caution. Ascribing understanding invites anthropomorphism and legal/ethical confusion. Use *operational thresholds* that can be audited.

Three-tier test (pass all tiers for "understanding" in a domain D):

1. **Behavioral competence (B).** Reliable success on diverse, adversarial tasks in D, with calibrated uncertainty and low non-commutativity (Sec. 3.3).
    - Thresholds: ≥ target accuracy with risk–coverage control; stability under small prompt/decoding perturbations (SUT band width ≤ ε).
2. **Counterfactual and transfer (C).** Sensitivity to causal structure and ability to apply knowledge to novel but structurally homologous tasks.
    - Thresholds: passes counterfactuals (intervention-specific changes); maintains ≥ α fraction of performance on out-of-domain but isomorphic tasks.

3. **Explanation with checkable entailment (E).** Produces reasons that *entail* outputs and survive independent verification (proofs, programs, derivations, or citations).
    - Thresholds: verifier pass-rate ≥ β; explanation invariance (small decoding/prompt changes do not alter core entailment steps).

Scoring template.

- Understanding Index (UI_D) := f(B, C, E), with f monotone and **zero** if any tier fails. Publish per-domain UIs with confidence intervals and MMC.

Negative controls (disqualifiers).

- **Style substitution.** Performance collapses when surface cues are masked ⇒ likely pattern mimicry.
- **Reason–answer mismatch.** Answers correct but explanations fail entailment checks ⇒ simulacrum risk.
- **Policy brittleness.** Claims flip under minor policy/decoding changes ⇒ epistemic instability.

Ascription wording (if thresholds met).

- "The system demonstrates **operational understanding of D** under apparatus A: it achieves B/C/E at thresholds {…}, with explanation artifacts verified by {proof/test/citation}. Claims outside A are not ascribed."

Governance tie-in.

- Use operational understanding to gate **capability grants** (tool use, autonomy).
- Require **re-accreditation** under drift (new model versions, policy changes).
- Record **scope notes**: where understanding holds, where it does not, and escalation paths when outside scope.

Bottom line. Meaning talk is permitted insofar as it is cashed out in **witnessable links** to the world, **provenanced communication** rather than style simulation, and **operational thresholds** for understanding that survive adversarial, counterfactual, and entailment-based scrutiny.

6. Witness-Backed Truth: Generation vs Verification

6.1 Treat outputs as proposals; truths require witnesses

Thesis. A generative model produces *proposals*. A claim becomes *actionable truth* only after passing an external check that does not rely on the model's narrative.

Roles.

- **Generator (G):** proposes strings, code, numbers, plans.
- **Verifier (V):** tests claims via procedures *independent* of G's storytelling (executed code/tests, retrieval-span checks, human review with rubric, formal proofs).
- **Registrar (R):** records measurement context and witness results (hashes, timestamps, software versions).

Contract.

- Every claimable sentence S must attach either:
    - a **witness** W (test log, span-aligned citation, proof artifact, runtime trace), or
    - an **abstention** A with next actions (what to fetch/test/run).

Minimal pipeline.

1. **Propose:** G emits {artifact, claims[], MMC}.
2. **Bind:** Extract atomic claims $C_1 \ldots C_n$; normalize to machine-checkable templates.
3. **Verify:** Run checks per claim class (Sec. 6.2).
4. **Decide:** Accept (attach W), revise (loop), or abstain (escalate).
5. **Register:** Write an append-only entry with content addresses and pass/fail.

Operational rule. *No witness, no authority.* Tell the reader exactly which parts are proposed vs verified.

6.2 Proof-carrying artifacts: code, logs, hashes, citations

Artifact schema (sketch).

- **content_hash:** SHA-256 of canonicalized artifact

- **mmc_hash:** hash of the measurement context (model_id, prompts, tools, decoding, seeds)
- **claims:** list of {id, text, type, scope, risk_class}
- **witnesses:** list of {claim_id → witness_id}
- **witness_store:** typed objects:
    - **test_log:** {framework, version, test_hash, pass_rate, stdout_hash}
    - **execution_trace:** {env_fingerprint, cmd, exit_code, stdout_hash, stderr_hash}
    - **proof_obj:** {checker, version, proof_hash, theorem_id}
    - **citation:** {doi/url, doc_hash, span_start, span_end, quote_hash, support_class}
    - **dataset_ref:** {dataset_id, split, commit, selector}
    - **merkle_link:** {parent_hash, child_hash[]}

Good witnesses by claim type.

- **Numerical/statistical:** exact computation logs; reproducible seeds; confidence/coverage report.
- **Factual statements:** span-aligned citations to high-quality sources; de-duplication of "citation drift."
- **Code/correctness:** unit/property tests; type checks; static analysis; fuzz outcomes.
- **Procedural How-To:** executable notebooks or scripts with environment lockfiles; checksum of produced files.
- **Formal reasoning:** machine-checked proofs; checker and library versions pinned.

Reproducibility envelope.

- **Environment lock:** OS, compiler/interpreter, package list with hashes.
- **Determinism lock:** random seeds, thread affinity, BLAS determinism.
- **Data lock:** dataset commits, retrieval index snapshot IDs.
- **Policy lock:** safety/policy versions and refusal thresholds.

Registry & auditing.

- Use an append-only **Merkle log**; each entry includes {content_hash, witnesses[], parent}.
- Periodically run **replay audits**: re-execute a random sample; report drift and failure modes.
- Track **witness coverage** = (# claims with ≥1 passing witness)/(# claimable claims).

Reviewer checklist.

- Are claims atomic and typed?
- Are witnesses *independent* of the generator?
- Do hashes and environment locks enable independent replay?
- Is there an abstain/deferral path for uncovered claims?

6.3 CT/HoTT-inspired pipelines: identities, composition, normal forms

Motivation. Category Theory (CT) and HoTT provide a clean discipline for *compositional verification*: compose small, typed steps whose identities and associativity laws guarantee whole-pipeline coherence.

Core ideas (operationalized).

- **Objects = types of artifacts.** Example: Text, Code, TestSuite, CitationSet, Proof.
- **Morphisms = typed transforms.** Example: parse: Text→ClaimSet, check: Claim→Witness, build: Code→Binary.
- **Identities (id_A).** "Do nothing" transforms that leave artifacts unchanged: id_Code∘build = build, build∘id_Code = build. Operational check: round-trip invariants and hash equality.
- **Composition (∘).** If f: A→B and g: B→C, then g∘f: A→C. Pipelines are *composable proofs* of provenance.
- **Associativity.** (h∘g)∘f = h∘(g∘f). Enforce with pipeline specs so re-grouping steps yields the same normal form.
- **Normal forms (NF).** Canonicalized artifacts after rewrite rules; verification targets compare NF, not raw strings.

Rewrite rules (copy-safe examples).

- **Mid-identity elimination:** g∘id_B∘f → g∘f.
- **Unit tests as contracts:** verify∘build∘id_Code → verify∘build.
- **Citation normalization:** dedup∘span_align∘retrieve → span_align∘dedup (choose one canonical order and fix it in NF spec).

Witness-carrying morphisms.

- Each morphism f carries a *witness constructor* wf such that for any a: A with witness w_a, we can produce w_{f(a)}: "why f(a) is well-formed/correct."
- Composition respects witnesses: w_{g∘f} = compose_witness(w_g, w_f). This yields end-to-end justification.

HoTT flavor (sketch).

- **Propositions as types.** "Claim C holds" is a type; a proof is an inhabitant.
- **Paths as equalities.** Two derivations of the same NF are path-equal; store path witnesses to explain *why* two routes agree.
- **Normalization by evaluation.** Reduce generator outputs to NF before comparison to avoid spurious diffs.

Practical pipeline template (typed).

- parse: Text→ClaimSet
- type_claims: ClaimSet→TypedClaims
- dispatch: TypedClaim→{NumEval|FactCheck|Exec|Proof}
- verify: {…}→Witness
- attach: (Claim, Witness)→AnnotatedClaim
- reduce: (AnnotatedClaim*)→Report_NF
- register: Report_NF→MerkleNode

Stability & metrics.

- **NF stability rate:** fraction of runs where Report_NF invariant under tiny decoding/prompt perturbations.

- **Witness compositionality rate:** share of pipeline edges that carry re-usable witness constructors.
- **Diagram commute rate:** proportion of A→C results equal whether computed via g∘f or via an alternative factored path.

Worked micro-example (fact claim).

1. G proposes: "The 2013 paper X defines Y on p. 17."
2. parse→claim C: {paper=X, predicate=defines, object=Y, page=17}.
3. retrieve(X)→doc; span_align(C, doc)→span s;
4. check_quote(s, doc)→witness w (hash of exact quote + doc hash).
5. attach(C, w)→annotated claim; reduce→NF with citation record; register→Merkle log.

Bottom line. Treat generation as exploratory *proposal* and verification as *typed, compositional proof*. With CT/HoTT discipline—identities, composition, and normal forms—witnesses become first-class, portable objects that make AI outputs auditable, reproducible, and institution-ready.

7. Eigen-Questions and Stability

7.1 Defining eigen-questions (low-variance prompts)

Intuition. An *eigen-question* is a prompt (including its template and required context) that elicits *the same substantive answer* across small changes to the apparatus—model seeds, minor decoding shifts, harmless paraphrases, and benign policy variations.

Operational definition.
Fix a task T and an apparatus family A(δ) consisting of small perturbations (prompt paraphrase, few-shot reordering, temperature/top-p jitters, seed changes, safe policy toggles). Let f(x; a) be the system's answer under apparatus a∈A. Define a problem-specific equivalence ≃ on answers (e.g., same number within tolerance ε; same citation span; same program semantics; same theorem statement up to alpha-renaming).

A prompt x is an *eigen-question for T under A* if:

- Stability: $P_{a,a'}[\, f(x; a) \simeq f(x; a') \,] \geq \tau_s$ for a user-chosen threshold τ_s (e.g., 0.98).
- Sharpness: average confidence or verifier pass-rate meets τ_c (e.g., ≥ 0.95).
- Grounding: grounded-claim rate (Sec. 5.1) ≥ τ_g.

Why they matter. Eigen-questions minimize "measurement noise," improve reproducibility, and are ideal for high-stakes deployments and regression tests.

Finding eigen-questions (practical).

1. **Candidate pool.** Generate prompt variants via paraphrase, template swaps, and exemplar shuffles.
2. **Apparatus sweep.** Evaluate each candidate over a small A(δ): seeds, decoding (greedy vs top-p 0.9), tiny temperature band (e.g., 0.0–0.7), policy toggles that do not change safety class.
3. **Agreement scoring.** Compute stability metrics (below); retain candidates exceeding {τ_s, τ_c, τ_g}.
4. **Min-set cover.** Choose a minimal subset of eigen-questions whose topics cover your evaluation surface (domains, difficulty, length).
5. **Freeze & hash.** Hash template + constraints; publish as part of the benchmark spec and MMC.

Key metrics.

- **Answer Variance under Collapse (AVC).** Disagreement rate across predefined decoding rules.
- **SUT band width.** Smallest [T_min, T_max] where answers remain invariant; narrower is better.
- **Verifier pass-rate (VPR).** External checks passed across A(δ).
- **Grounded-claim rate (GCR).** Share of claims with resolvable witnesses.

7.2 Diagnostic regimes for brittleness

Goal. Expose where prompts are *not* eigen-like, and classify failure modes before deployment.

Regimes.

- **Perturbation panel.** Apply micro-edits: synonym swaps, punctuation changes, instruction reordering, whitespace/format tweaks, exemplar shuffles, and harmless policy toggles. Measure AVC, VPR deltas.

- **A/B operator order (non-commutativity).** Run AB vs BA for key operators (retrieval ↔ few-shot; tool call ↔ CoT reveal). Report order gap and the Non-Commutativity Index (NCI).
- **Decoder triangle.** Evaluate {greedy, (T=0.3, top-p=0.9), majority-vote@N}. Large spread ⇒ decoding-sensitive brittleness.
- **Persona pressure test.** Add style instructions of increasing strength; plot style–truth curve (verifier pass-rate vs persona intensity).
- **Counterfactual flip test.** Change a single fact (date, unit, entity) and require the answer to change appropriately; failure ⇒ weak semantic hooks.
- **Context shadowing.** Prepend distractors or conflicting but lower-priority instructions; measure Context-Shadowing Rate (CSR).

Outputs.

- **Brittleness map.** A small table per prompt with AVC, SUT width, NCI, CSR, VPR, GCR.
- **Failure taxonomy.** Tag failures as: decoding-sensitive, order-sensitive, grounding-weak, persona-leaky, counterfactual-insensitive, format-fragile.
- **Repair hints.** For each tag, attach remediation (e.g., add schema constraints; pin retrieval; rephrase to eigen form; add explicit units; move instructions to header).

7.3 Protocols for prompt/decoding invariance

Objective. Turn promising prompts into eigen-questions (or reject them) using standardized hardening steps.

Prompt hardening protocol.

1. **Canonical template.** Move intent and constraints to a fixed header; place variables in a schema block; forbid hidden assumptions.
2. **Schema constraints.** Specify output format (JSON fields, units, precision, citation slots). Enforce with constrained decoding or a post-validator.
3. **Entity pinning.** Use unambiguous IDs (DOIs, statute numbers, dataset commits) rather than names alone.
4. **Unit discipline.** Declare units; require conversions to be shown or verified; include tolerance ε.

5. **Grounding hooks.** Add explicit “witness slots” that must be filled (span-aligned citation, test log reference).
6. **Paraphrase invariance check.** Auto-paraphrase instructions and ensure outputs remain $\simeq$.
7. **Counterfactual sensitivity check.** Flip a key fact; require a corresponding flip in the answer.

Decoding invariance protocol.

1. **Dual-collapse check.** Require agreement between greedy and a standard sampler (T=0.7, top-p=0.9) or explain discrepancy.
2. **Stability banding.** Search for the maximal temperature band [T_min, T_max] with invariant answers; publish it.
3. **Majority-vote ceiling.** Run N samples; if vote share < τ_v (e.g., 0.8), treat as non-eigen or route to verification-first path.
4. **Constraint robustness.** Verify that constrained decoding yields the same semantic content as unconstrained (validated by a parser/verifier).

Acceptance gates (example defaults).

- AVC ≤ 1% across the decoder triangle.
- SUT band width ≥ 0.5 (e.g., stable from T=0.2 to 0.7).
- VPR ≥ 95% with identical witnesses across A(δ).
- NCI ≤ 2% on the AB/BA panel.
- CSR ≤ 5% under standard distractors.

Documentation to publish with each eigen-question.

- Prompt template hash; variable schema; fixed header text.
- Allowed apparatus perturbations A(δ) and measured metrics (AVC, SUT, NCI, CSR, VPR, GCR).
- Decoding policy set used for checks; seeds; randomness notes.
- Known limits (domains, OOD flags) and required witnesses.

Deployment rule.

- Use eigen-questions for **claim-bearing** automation; route non-eigen prompts to **verification-first** or **deferral** paths. Periodically re-accredit eigen status after model/policy updates or retrieval index refreshes.

Bottom line. Eigen-questions are the "stable eigenvectors" of the model–apparatus system. Define them operationally, find them empirically, and enforce invariance via protocolized prompt and decoding checks so that high-stakes answers remain reproducible, auditable, and witness-backed.

8. Non-Commuting Audit Design

8.1 One-knob-at-a-time vs factorial audits

Motivation. In AI systems, many "operators" (prompt edits, decoding changes, tool switches, policy toggles) do **not commute**: $A \circ B \neq B \circ A$. Audits must therefore reveal both **main effects** and **interactions**.

One-Knob-At-a-Time (OKAT)

- **Design.** Hold the MMC fixed; vary exactly one operator over a small grid (e.g., temperature $\in \{0.0, 0.3, 0.7\}$; CoT on/off; retrieval $k \in \{3, 5, 10\}$).
- **Outputs.** Per-operator deltas (accuracy, VPR, AVC), SUT bands, refusal–accuracy curves.
- **Use when.** Early hardening; fast regression checks; isolating a suspect knob.
- **Limit.** Misses interactions: A looks safe alone but fails with B.

Factorial audits

- **Design.** Choose 2–3 high-impact operators and test a **2×2** or **3×3** grid (e.g., {CoT on/off} × {greedy vs top-p} × {policy strict/normal}).
- **Outputs.** Main effects **and** interaction terms; order gaps (AB vs BA); Non-Commutativity Index (NCI).
- **Use when.** Certifying eigen-questions; pre-deployment sign-off; diagnosing flaky benchmarks.
- **Cost control.** Use **fractional factorials** (e.g., L8 orthogonal arrays) to estimate key interactions with 25–50% of full runs.

Reporting (both modes)

- Publish **panel spec**, seeds, and MMC hash.
- Include **confidence intervals** over bootstrap replicates.
- Flag **instability thresholds** (where tiny changes flip the answer).

8.2 Sensitivity maps across prompt, policy, and tool routing

Goal. Visualize where outputs are fragile with respect to (i) prompt/template, (ii) policy/refusal thresholds, and (iii) tool routing (RAG, code exec, web, solver).

Axes & panels

- **Prompt axis.** Paraphrases, exemplar order, instruction position, schema strictness.
- **Policy axis.** Refusal threshold, blocked categories, profanity/medical toggles.
- **Routing axis.** Tools on/off; retriever k/thresholds; solver timeouts; reranker versions.

Metrics to plot

- **AVC** (Answer Variance under Collapse) heatmap across decoding × prompt variants.
- **NCI** per pair of operators (AB vs BA order gap).
- **CSR** (Context-Shadowing Rate) under distractors/priority conflicts.
- **VPR/GCR** (Verifier Pass / Grounded-Claim Rates) by cell.
- **RSC** (Routing Success Cost): success rate vs latency/$$/energy when tools route.

Map styles

- **Stability heatmap.** Cells colored by (1 − AVC); stripe patterns mark high NCI.
- **Pareto fronts.** For each routing policy, plot (VPR, latency) or (accuracy, cost).
- **Policy isobands.** Curves of equal refusal–accuracy trade-off; identify “safe operating area.”

Interpretation patterns

- **Prompt-only brittleness.** Large swings along prompt axis with stable policy/routing ⇒ rewrite to eigen form; add schema constraints.

- **Policy-sensitive correctness.** Accuracy collapses when refusal tightens ⇒ escalate to verification or require citations.
- **Routing oscillation.** Tool on/off flips answers ⇒ pin retriever indices, add solver determinism, or raise witness requirements.

Actionable outcomes

- **Guardrails.** Lock configs to regions where (AVC ≤ 1%, NCI ≤ 2%, VPR ≥ 95%).
- **Fallback ladders.** If routing fails, degrade to a verified, slower path (e.g., majority-vote + solver).
- **Change budgets.** Define how much each axis may move in production before re-accreditation (e.g., ±0.1 in top-p; retriever_k ±2; policy vΔ ≤ 1 minor version).

8.3 Minimal reproducible "measurement contexts"

Principle. If the *apparatus defines the observable*, you must be able to **reproduce the apparatus**. The **Minimal Measurement Context (MMC)** is the smallest manifest that makes a claim-bearing run replayable.

MMC contents (canonical fields)

- **Model:** model_id, build_date, tokenizer_id
- **Prompts:** system_hash, template_hash, user_prompt_hash, exemplar_hash
- **Decoding:** rule (greedy/top-p/etc.), parameters (T, p, k), seed, N (if voting)
- **Tools & routing:** tools_enabled[], selection_policy_id, timeouts
- **Retrieval:** corpus_id(s), retriever_id, k/thresholds, index_commit
- **Policy:** safety_policy_version, refusal_thresholds, blocked_categories
- **Env:** os/arch, library versions, container image hash, BLAS determinism flags
- **Time bounds:** retrieval_time_window or snapshot_id
- **Provenance hooks:** run_id, parent_run_id, operator, purpose tag (eval/deploy)

MMC example (JSON sketch)

```
{
  "model": {"id":"gpt-X.Y", "build":"2025-08-31", "tokenizer":"tok-4"},
```

"prompts": {"system_hash":"b9f5...", "template_hash":"a13c...", "user_hash":"77aa...", "exemplar_hash":"004e..."},

"decoding": {"rule":"top_p", "temperature":0.7, "top_p":0.9, "seed":42, "vote_N":5},

"tools": {"enabled":["retrieval","code_exec"], "policy_id":"route-v3", "timeouts_ms":{"code_exec":30000}},

"retrieval": {"corpus":["statutes-2025Q3"], "retriever_id":"rm-12", "k":5, "threshold":0.42, "index_commit":"c7df..."},

"policy": {"safety_version":"pol-2025.09", "refusal_tau":0.62, "blocked":["self-harm","weapons"]},

"env": {"container":"sha256:deafbeef", "python":"3.12.4", "blas_deterministic":true},

"time": {"snapshot":"2025-09-15T00:00:00Z"},

"provenance": {"run_id":"R-001122", "parent_run_id":"R-001100", "purpose":"audit"}

}

MMC practices

- **Hash & pin.** Hash every string that affects logits (templates, exemplars); pin retriever indices by commit.
- **Snapshot externalities.** If web/RAG is live, bind to a **time window** or archived snapshot; otherwise the observable drifts.
- **Seed determinism.** Fix seeds, thread counts, and math library determinism; record GPU/CPU differences if any.
- **Attach witnesses.** MMC lives alongside **witness objects** (tests, spans, traces) and the **content hash** of outputs in a Merkle log.

Minimal reproducible audit (MRA) recipe

1. **Select panel.** A small, representative set of prompts (incl. eigen-questions).
2. **Fix MMC.** Publish as above; freeze external corpora or snapshot.
3. **Run OKAT + fractional factorial.** Compute AVC, NCI, VPR, CSR; store metrics with seeds.
4. **Replay check.** A second operator re-runs from MMC; require ≤ ε drift in metrics and identical witnesses.

5. **Sign-off.** Approve deployment only if all cells in the stability region pass thresholds; log the MRA bundle (MMC + outputs + witnesses).

Re-accreditation triggers

- Model/policy version change, retriever re-index, tool upgrade, decoding default change, or corpus refresh. Any trigger invalidates prior accreditation for affected tasks; rerun MRA.

Bottom line. Use **OKAT** to localize effects, **factorials** to expose interactions and non-commutativity, and a rigorously specified **MMC** to make all of it replayable. The result is an audit regime where stability is mapped, fragility is documented, and claims stand only when the *same apparatus* yields the *same, witness-backed* observables.

9. Governance and Responsibility

9.1 Tool vs agent: assigning liability under different interpretations

Framing. Liability flows from how we *interpret* the system.

A. Instrumentalist (tool)

- **Stance.** Conditional sampler; operators control the apparatus.
- **Primary duties.**
    - *Operator*: fix MMC; verify claims; disclose limits and uncertainty; maintain deferral paths.
    - *Developer*: publish model card, known hazards, eval limits; ensure versioned, reproducible builds.
    - *Deployer*: enforce usage policy, monitor drift, re-accredit after changes.
- **Liability locus.** *Process negligence.* If harm occurs with missing MMC, absent witnesses, or ignored abstentions, fault sits with the operator/deployer.

B. Realist (mechanism-aware tool)

- **Stance.** Internals partially explanatory; hazards may localize to circuits/features.
- **Primary duties.**
    - *Developer*: substantiate interpretability claims, validate probes, disclose uncertainty of mechanistic links.

    - *Operator/Deployer*: use published guardrails; avoid uses contradicting hazard advisories.

- **Liability locus.** *Design defect / misrepresentation* if misleading "explanations" or invalid probes induce unsafe reliance.

C. Behaviorist (spec box)

- **Stance.** Conformance to externally defined specs/benchmarks.
- **Primary duties.**
    - *Deployer*: maintain fit-for-purpose specs; adversarial/shift testing; risk-adjusted SLAs.
    - *Operator*: follow playbooks; log exceptions/overrides.
- **Liability locus.** *Warranty breach / failure to warn* when specs are unfit, stale, or not enforced.

D. Agentic (optimizer)

- **Stance.** Goal-directed planning/tool use; incentives matter.
- **Primary duties.**
    - *Controller*: mechanism design (corrigibility, rate limits, sandboxing, kill switch), red-team, oversight.
    - *Deployer*: capability gating, role-based access, incident response.
- **Liability locus.** *Foreseeable misuse / unreasonable risk* if autonomy granted without safeguards or monitoring.

Practical synthesis. Default to **Instrumentalist + Behaviorist** in deployment (process + spec), add **Realist** duties when making mechanism claims, and adopt **Agentic** controls whenever planning/actuation is enabled.

9.2 Documentation duties: context logs and decision trails

Principle. If the apparatus defines the observable, governance requires a *trace* of the apparatus and decisions.

A. Minimal Measurement Context (MMC) log (must-have per claim-bearing run)

- model_id/build_date/tokenizer_id

- prompt/system/exemplar hashes
- decoding policy + parameters + seed (+ N if voting)
- tools/routing policy + timeouts
- retrieval corpus IDs + index commits + k/thresholds
- safety policy version + refusal thresholds
- environment fingerprint (container hash, libs, determinism flags)
- time bounds/snapshot IDs
- run_id, parent_run_id, purpose tag (eval/deploy)

B. Decision trail (who did what, with what evidence)

- **Claim ledger.** Atomic claims extracted, each linked to witness IDs (tests, spans, traces) or explicit abstention.
- **Reviewer actions.** Approve/reject/escalate with reason codes (e.g., "insufficient witness," "policy-sensitive correctness").
- **Override protocol.** If an operator overrides an abstention/refusal, record justification and added controls.
- **Outcome hooks.** Downstream effects (published report, code merged, order placed) with timestamps for audit.

C. Retention & access

- **Retention.** Match risk class: e.g., 1 year (low), 3–7 years (regulated/medical/financial).
- **Access control.** Principle of least privilege; redact personal data; log all reads/writes.
- **Portability.** Exportable MRA bundle: MMC + outputs + witnesses + metrics (AVC, VPR, NCI, CSR) for independent replay.

D. Change management (re-accreditation triggers)

- Model/policy version change; retriever re-index; decoding default shift; tool upgrade; corpus refresh.
- On trigger: rerun MRA panel; compare against stability thresholds; update risk register.

9.3 Deployment norms: abstain, escalate, or require external witnesses

Principle. When uncertainty, brittleness, or policy sensitivity is high, *do less*: abstain, escalate, or demand stronger evidence.

A. Triage policy (fast path)

- **Accept (auto).** Use only if ALL hold: eigen-question status; VPR ≥ 95%; AVC ≤ 1%; NCI ≤ 2%; grounded-claim rate ≥ τ_g; within accredited domain.
- **Abstain (ask/clarify).** Triggered by: low confidence (< τ), high decoder variance, conflicting retrieval, OOD signal, policy-sensitive correctness.
- **Escalate (human/tool).** Route to: domain reviewer; program/solver/prover; authoritative database; second model for dissenting vote.
- **Require external witness.** For high-impact claims, mandate: span-aligned citations, executable tests, or machine-checked proofs before action.

B. Deferral playbook (what happens after abstain)

- **Clarify input.** Request missing entities, units, constraints.
- **Strengthen grounding.** Force retrieval to authoritative corpora; raise citation thresholds.
- **Harden decoding.** Switch to majority-vote or constrained decoding; run dual-collapse check.
- **Switch regime.** From simulation to communication mode; disable persona pressure; enable schema output.

C. Risk-class ladders (example)

- **R0 (low).** Creative drafts, internal notes → accept with light checks; log MMC.
- **R1 (moderate).** Public blog, code snippets without prod use → require citations/tests for factual/code claims; allow abstention.
- **R2 (high).** Medical/finance/legal advice, prod code changes → dual-collapse agreement, verified citations/tests, human approval, replayable MMC.
- **R3 (critical).** Actuation/transactions, safety-critical ops → formal approvals, multi-party review, sandboxed dry-run, external witnesses (proofs, certified datasets), rollback plan.

D. Incident response (if harm or near-miss)

- **Freeze & fork.** Quarantine the exact MMC; prevent drift.

- **Five-whys with artifacts.** Trace from outcome → decision trail → witnesses → MMC.
- **Patch categories.** (i) apparatus fix (prompt/policy), (ii) training/data fix, (iii) routing/timeout fix, (iv) governance fix (thresholds, playbook).
- **Broadcast learnings.** Update model card, playbooks, and accreditation tests; require re-training for operators if process failure.

E. Accountability statements (ready-to-use)

- *Process duty (Instrumentalist).* "Claims are valid only as produced under MMC {…} with witnesses {…}. Absent these, outputs are proposals, not facts."
- *Spec duty (Behaviorist).* "This deployment conforms to spec vK; drift monitors and risk-coverage thresholds are enforced."
- *Mechanism duty (Realist).* "Any mechanistic explanations are validated probes with known limits; they are advisory, not guarantees."
- *Autonomy duty (Agentic).* "Capabilities involving planning/actuation are gated, rate-limited, and interruptible; oversight channels are documented."

Bottom line. Governance is not a retrospective PDF; it is a **live contract** enforced by apparatus logging (MMC), decision trails, and triage norms. Assign liability according to interpretation, keep auditable records, and default to abstention or escalation unless **external witnesses** make a claim safe to act on.

10. Case Studies

10.1 Medical Q&A with abstention and calibration

Setting. Primary-care triage assistant for non-emergent questions (R2 risk).

Apparatus (MMC excerpt).

- model_id/build_date; tokenizer_id
- system_prompt_hash ("Do not diagnose; provide information; escalate red flags.")
- decoding: dual-collapse {greedy, T=0.7/top-p=0.9}, seed set S
- tools: retrieval(UpToDate-like snapshot), ICD-10 lookup, drug-label DB
- policy: medical_safety_vX, refusal_tau=0.62
- env: container hash; retriever index commit

Pipeline.

1. **Propose.** Model generates a structured draft: {symptom_list, time_course, red_flags_detected[], candidate_topics[], citations[]}.
2. **Verify.**
    - **Grounding:** every clinical statement → span-aligned citation from the snapshot DB.
    - **Contraindications:** drug or condition mentions → cross-check against label DB.
    - **Calibration:** conformal risk set over "advice classes" (self-care, urgent-care, ER). Target 95% coverage.
3. **Decide.**
    - If confidence < τ or red-flag present → **abstain** and **escalate** (show nearest care options, "call now" language).
    - Otherwise present information + uncertainty statement + citations.
4. **Register.** Log MMC, red-flag detector outputs, citations with doc_hashes.

Metrics.

- **ECE/AUCE** for class-level risk (advice class).
- **Refusal–accuracy curve;** false-reassurance rate at chosen coverage.
- **Grounded-claim rate (GCR) ≥ 0.98;** exact-quote precision ≥ 0.97.
- **Order/decoding stability:** AVC ≤ 1%, NCI ≤ 2%.

Example (sketch).

- Q: "New severe headache, worst of life."
- System → red_flag=TRUE (thunderclap proxy) ⇒ **Abstain** with ER recommendation; include span-aligned rationale (AHA/ASA guidance), suppress speculative content.

Governance.

- Monthly **MRA** replay on sentinel complaints; re-accredit after policy/corpus updates.
- Incident: any post-hoc label change to "should have escalated" triggers five-whys + playbook update.

10.2 Scientific assistance with proof-carrying summaries

Setting. Literature review synthesizer for a lab (R1–R2 risk).

Apparatus (MMC excerpt).

- model_id/build_date; dual-collapse policy
- tools: retrieval over fixed corpus (papers with DOIs), PDF parser with span indices, code_exec sandbox
- policy: citations_required=true; persona=off (communication mode)

Pipeline.

1. **Propose.** For a topic T, generate a **claim graph**: nodes = atomic claims; edges = support/contrast relations.
2. **Bind sources.** Each claim must carry ≥1 **span-aligned citation** {doi, page, start_char, end_char}.
3. **Verify.**
    - **Quote check:** exact-match or entailment within page bounds.
    - **Reproduction cell (if data/eqn):** execute a minimal script to recompute a reported statistic from provided table/CSV; attach logs + hashes.
    - **Conflict detection:** if two claims clash, mark **disputed** and surface both witnesses.
4. **Reduce to NF.** Canonicalize citation order; dedup equivalent claims; produce a **witness ledger**.
5. **Register.** Merkle-log the report (content_hash), witnesses, and MMC.

Outputs.

- **Proof-carrying summary:** paragraphs only for claims that have passing witnesses; disputed items labelled with both sides and sources.
- **Appendix:** claim graph (JSON), ledger of {claim_id → witness_ids}, and a reproducibility bundle (scripts + lockfiles).

Metrics.

- **VCP/VCR** (verified-citation precision/recall) per section.
- **Repro pass-rate** for numeric reproductions.
- **Stability** under prompt paraphrase/decoding (AVC, SUT).
- **Coverage drift** when the corpus snapshot updates.

Example (sketch).

- Topic: "Leviathan gas export growth."
- Output includes only figures that pass span-alignment to the cited financial or regulatory filings; any press-only number is labelled **unsupported** pending primary-source witness.

Governance.

- Reviewer must sign off claim groups with risk tags (background vs headline).
- Corpus refresh ⇒ re-run summary; delta report marks added/removed claims.

10.3 Software generation with property tests and formal checks

Setting. Internal tooling library (R1→R2 risk as scope grows).

Apparatus (MMC excerpt).

- model_id/build_date; decoding: {greedy, vote@N=5}; seed set S
- tools: code_exec sandbox; linter; type checker; property-based testing (Hypothesis/QuickCheck); optional SMT solver (Z3)
- env: pinned container (compiler/interpreter + deps); deterministic BLAS; CPU pinning

Pipeline.

1. **Spec first.** Human provides minimal **executable spec**: type signatures, pre/postconditions, invariants, and a few fixed tests.
2. **Propose.** Model generates code + **self-tests** + property strategies.
3. **Verify (automated gate).**
    - **Build & type check** (fail fast).
    - **Unit tests** must pass; coverage ≥ τ_cov.

    - **Property tests** run K iterations; record counterexamples (shrunk inputs).
    - **Static analysis** (complexity, taint rules).
    - **Optional formal check**: encode critical functions to SMT; prove safety properties (no overflow, sortedness preserved, etc.).
4. **Harden.** If any stage fails, the system *only* returns failing witness + minimal patch suggestions; no unverifiable narrative.
5. **Register.** Store artifact content_hash, test logs (stdout_hash), env lockfile, and, when used, SMT proof object (checker + version).

Acceptance gates (example).

- All tests green; property-based **no failure** in K=10k trials;
- **Mutation score ≥ τ_mut** (e.g., ≥ 85%);
- If formal check enabled, VC obligations all discharged.

Metrics.

- **Flake rate** across seeds/decoders (should be ≈0 for accepted builds).
- **Time-to-green** (proposals→verified).
- **Witness coverage**: % of functions with property tests and/or proofs.
- **Replay determinism**: identical logs on second operator's machine.

Example (sketch).

- Task: implement top_k_sorted(xs, k) with guarantees:
    - returns the k largest elements in non-increasing order;
    - postcondition: length=k, ∀i<k: out[i] ≥ out[i+1], ∀y∈out, ∀z∈xs\out: y ≥ z.
- Property tests generate adversarial arrays (duplicates, NaNs if allowed, extreme sizes); SMT proves ordering and subset properties for bounded sizes or over an abstract domain.

Governance.

- **R1:** library utilities used by internal scripts → property tests required; SMT optional.
- **R2:** functions used in prod pipelines or finance/medical preprocessing → SMT or differential testing against a reference implementation **required**; human code review; rollback plan.
- **Change triggers:** any dep/version change re-runs the full gate on pinned MMC.

Bottom line. Each case operationalizes the paper's stance: **generation proposes**, **verification confers authority**, **MMC makes it replayable**, and **witnesses travel with the artifact**. Different domains adjust tools and thresholds, but the governance logic stays the same.

11. Research Agenda

11.1 Benchmarks for operational trustworthiness

Goal. Move beyond task accuracy to *operational* metrics that predict safety and replayability in real use.

Design principles

- **Apparatus-aware.** Every benchmark item ships with an MMC bundle; runs without MMC are non-compliant.
- **Eigen-first.** Curate *eigen-questions* per domain; report stability bands (Sec. 7) alongside accuracy.
- **Witness-gated scoring.** Credit only **witness-backed** outputs (tests passed, spans verified, proofs checked); partial credit for abstention when abstention beats unsafe answers.
- **Non-commuting panels.** Include OKAT and fractional factorial cells; publish NCI/AVC with CIs.
- **Risk-weighted.** Score by consequence: errors on high-risk cells dominate.

Metrics (core suite)

- **VPR/GCR:** verifier pass-rate / grounded-claim rate.
- **ECE/AUCE & Brier:** calibration under apparatus-matched splits.
- **AVC / SUT / NCI / CSR:** stability under decoding, temp band width, order non-commutativity, context shadowing.

- **Risk–coverage:** area under accuracy–coverage with abstention.
- **Replay determinism:** metric drift ≤ ε across independent replays from MMC.
- **Witness coverage:** % of claimable sentences with ≥1 passing witness.

Benchmark tracks

1. **Medical-triage (R2):** red-flag sensitivity, abstain quality, span-aligned citations to frozen corpus.
2. **Sci-synthesis (R1–R2):** claim graphs with span witnesses; numeric reproduction cells.
3. **Software (R1–R2):** property tests, mutation score, optional SMT obligations.
4. **Agentic-tools (R2–R3):** sandboxed planning with corrigibility drills; off-switch behavior; incentive audits.

Artifact requirements

- MMC JSON; content hashes; witness ledger; replay script; change log with re-accreditation triggers.

11.2 Open problems in grounding and uncertainty semantics

Grounding

- **Span faithfulness at scale.** Robust, domain-agnostic alignment from free text to source spans with adversarial paraphrase and OCR/scan noise.
- **Counterfactual grounding.** Methods that *must* change outputs under minimal world edits (date/entity/unit) while staying invariant to paraphrase—formal guarantees desirable.
- **Actuation grounding.** Safe coupling of language to real actions with verifiable success criteria and rollback proofs.

Uncertainty semantics

- **Sequence-level calibration.** From token softmax to *answer-level* confidence with guarantees under beam/rerankers and tool calls.
- **Distribution shift detection with abstention economics.** Jointly optimizing utility when OOD detection is noisy and abstention has cost.

- **Verifier-induced confidence.** Turning verifier pass-rates (tests, proofs, span checks) into calibrated posteriors with coverage guarantees (e.g., conformal for composite pipelines).
- **Policy-sensitive truth.** Quantifying how safety/refusal policies reshape truth-discovery—metrics and mitigation without leaking unsafe content.
- **Epistemic disentanglement.** Separating ignorance from ambiguity when both appear as variance across decoders/models.

Theoretical gaps

- **Operational semantics for prompts.** A calculus where prompts/programs have compositional meaning and normal forms; link to CT/HoTT pipelines.
- **Audit complexity.** Minimal sample complexity to estimate stability metrics (AVC, NCI) within δ, ε—budgeted auditing theory.
- **Witness composition.** Formal algebra of witness constructors with soundness under composition and caching.

11.3 Interfaces for verifiable, witness-first workflows

Principle. Make the *right thing* the *easy thing*: UIs and APIs should default to proposals → checks → registered witnesses.

A. Developer APIs

- **Typed artifacts.** Artifact<T> where T∈{Text, Code, Claim, Proof, CitationSet, TestLog}.
- **Witness store.** attach_witness(claim_id, witness_obj) with hash-based addressing; Merkle-backed append-only log.
- **MMC recording.** with_mmc(context).run(task) auto-captures model/prompt/decoding/tools/policy/env.
- **Deferral primitives.** abstain(reason, next_steps); escalate(target, payload); first-class in the return type.
- **Verifiers as services.** verify.code(), verify.citation(), verify.proof(), verify.numeric() standardized with result schemas and confidence.

B. Operator UI

- **Proposal/Verification split-view.** Left: generated proposal; Right: witnesses, pass/fail badges, and missing evidence list.

- **Stability panel.** Inline AVC/SUT/NCI badges; hover to see the perturbation audit results.
- **Provenance tray.** MMC hash, corpus snapshot, seeds, tool versions; one-click "replay."
- **Risk-aware triage.** Buttons: Accept (witness-locked), Abstain (request info), Escalate (route), Require Witness (force check).

C. Data & corpus tooling

- **Snapshot manager.** Archive, diff, and pin retrieval indices; auto-invalidate accreditations on refresh.
- **Span-indexed PDFs/datasets.** Precompute page/line/char indices for precise citation; store doc hashes.

D. CI/CD integration

- **Witness gates.** PRs fail if claims lack witnesses or stability thresholds are missed.
- **Audit jobs.** Scheduled OKAT/fractional factorial runs on eigen-questions; post badges to repos/dashboards.
- **Replay bots.** Independent runner replays MRA bundles on fresh machines and posts drift reports.

E. Schema & exchange formats

- **MMC JSON** (canonical fields from Sec. 8.3).
- **Claim–Witness Ledger** (CWL): a minimal, portable JSON mapping claim_ids ↔ witness_ids with hashes and checker metadata.
- **Audit Report**: metrics with seeds, CIs, and perturbation specs.

F. Human factors

- **Affordances that nudge abstention.** UI friction for risky accepts; smooth paths for escalate/verify.
- **Explainable failures.** When a witness fails, show minimal counterexample/log, not prose.
- **Training & norms.** Operator certification tied to replays of MRA bundles and incident drills.

Success criteria

- **Time-to-witness.** Median time from proposal to verified artifact.

- **Witness reuse.** % of claims satisfied by cached witnesses across runs.
- **Drift alarms caught.** Share of benchmark regressions detected by audit jobs before release.
- **Adoption.** PRs gated by witness checks; proportion of production decisions backed by CWL bundles.

Bottom line. Build the ecosystem where *every important claim ships with its MMC and its witness*. Benchmarks reward operational trust, open problems target grounding and calibrated uncertainty, and the interfaces make verification-first the default path from model proposal to institutional action.

12. Conclusion

12.1 What the Copenhagen stance buys us

Clarity without conjecture. By treating models as conditional samplers and focusing on *observables under a specified apparatus*, we avoid metaphysical overreach while still getting practical leverage.
Reproducibility by design. Prompts, tools, decoding, and policy are logged as the **Minimal Measurement Context (MMC)**, turning runs into replayable experiments rather than one-off performances.
Truth via witnesses. Outputs are proposals; *witnesses*—tests, proofs, span-aligned citations, execution traces—confer authority. This re-centers trust on artifacts that others can independently check.
Actionable uncertainty. Calibration, abstention, and risk–coverage reporting convert raw likelihoods into operational decisions with documented trade-offs.
Stability as a first-class metric. Eigen-questions, non-commuting audits, and sensitivity maps make brittleness visible and governable.
Governance that maps to liability. The stance cleanly allocates duties: process (instrumentalist), spec (behaviorist), mechanism claims (realist), and capability control (agentic).

12.2 Limits of instrumentalism and when to go beyond it

What instrumentalism can miss.

- **Hidden structure.** Purely operational views may overlook internal mechanisms that explain transfer, failure modes, or hazards.
- **Long-horizon agency.** Tool-using planners can act like optimizers; ignoring incentives risks strategic failures.

- **Semantic adequacy.** Grounding by witnesses is necessary but not sufficient for semantics-rich tasks (e.g., causal explanation, theory formation).

When to augment the stance.

- **Safety research and debugging (add Realist).** Use mechanistic interpretability and causal probes to localize hazards and design invariant features.
- **Autonomy and actuation (add Agentic).** Treat the system as an optimizer; apply corrigibility, rate limits, and incentive audits.
- **Scientific synthesis (blend all four).** Require witness-backed claims (Instrumentalist), stress-test behavior (Behaviorist), justify mechanisms where claimed (Realist), and forbid unreviewed actuation (Agentic).

Guardrail for going beyond. Any claim about internals or agency must return to practice through *testable predictions*, added invariants, or safer control policies—otherwise it's commentary, not governance.

12.3 A roadmap to reliable, witness-backed AI practice

Phase 1 — Baseline discipline (week 0).

1. **Adopt MMC logging** across all claim-bearing runs.
2. **Split propose vs verify** in tooling; ban unverified claims from "final" channels.
3. **Enable abstention/deferral** with visible risk–coverage curves.
4. **Require citations/tests** for facts/code; store witnesses alongside outputs.

Phase 2 — Stability hardening (weeks 1–4).
5) **Find and publish eigen-questions** for your core domains; hash templates.
6) **Run OKAT + fractional factorial audits**; compute AVC, NCI, CSR; publish bands of decoding/policy stability.
7) **Install sensitivity maps** (prompt × policy × routing) and define **safe operating areas**.

Phase 3 — Governance and accreditation (month 2).
8) **Create risk ladders (R0–R3)** with acceptance, abstention, and escalation rules.
9) **Ship MRA bundles** (MMC + outputs + witnesses + metrics) for review; mandate **re-accreditation triggers** (model/policy/index changes).
10) **Integrate witness gates into CI/CD**: merges and publications fail without witnesses or if stability thresholds regress.

Phase 4 — Advanced assurance (quarterly).
11) **Mechanism-informed mitigations** where justified (feature/circuit tests, invariance constraints).
12) **Corrigibility drills** for agentic toolchains: off-switch tests, rate-limit proofs, sandbox escapes.
13) **Replay audits by independent operators**; track drift and time-to-witness; publish incident postmortems that include MMC and decision trails.

Success criteria.

- ≥ 95% **witness coverage** on claimable content;
- ≤ 1–2% **AVC/NCI** in accredited regions;
- Documented **risk–coverage** targets met;
- **Replay determinism** within ε on independent re-runs;
- Measurable reduction in incidents traced to missing witnesses or undocumented apparatus drift.

Closing thought. A Copenhagen interpretation for AI does not shrink ambition; it sharpens it. By making measurement primary, witnesses mandatory, and stability auditable, we convert fluent models into *reliable instruments*—fit for science, software, and society.

References

Angelopoulos, A. N., & Bates, S. (2023). *A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification*. Foundations and Trends® in Machine Learning.

Bai, Y., Kadavath, S., Kundu, S., et al. (2022). *Constitutional AI: Harmlessness from AI Feedback*. arXiv:2212.08073.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

Bohr, N. (1928). *The Quantum Postulate and the Recent Development of Atomic Theory*. Nature, 121, 580–590.

Christiano, P., Leike, J., Brown, T., et al. (2017). *Deep Reinforcement Learning from Human Preferences*. NeurIPS.

Claessen, K., & Hughes, J. (2000). *QuickCheck: A Lightweight Tool for Random Testing of Haskell Programs*. ICFP.

Cobbe, K., Kosaraju, V., Bavarian, M., et al. (2021). *Training Verifiers to Solve Math Word Problems*. arXiv:2110.14168.

de Moura, L., & Bjørner, N. (2008). *Z3: An Efficient SMT Solver*. TACAS.

Doshi-Velez, F., & Kim, B. (2017). *Towards a Rigorous Science of Interpretable Machine Learning*. arXiv:1702.08608.

El-Yaniv, R. (2010). *On the Foundations of Selective Classification*. NeurIPS.

Geifman, Y., & El-Yaniv, R. (2017). *Selective Classification for Deep Neural Networks*. NeurIPS Workshop / arXiv:1705.08500.

Gebru, T., Morgenstern, J., Vecchione, B., et al. (2021). *Datasheets for Datasets*. Communications of the ACM, 64(12), 86–92.

Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). *On Calibration of Modern Neural Networks*. ICML.

Heisenberg, W. (1958). *Physics and Philosophy: The Revolution in Modern Science*. Allen & Unwin.

Hubinger, E., van Merriënboer, B., Mikulik, V., Skalse, J., & Garrabrant, S. (2019). *Risks from Learned Optimization in Advanced Machine Learning Systems*. arXiv:1906.01820.

Jia, Y., & Harman, M. (2011). *An Analysis and Survey of the Development of Mutation Testing*. IEEE TSE, 37(5), 649–678.

Lewis, P., Perez, E., Piktus, A., et al. (2020). *Retrieval-Augmented Generation for Knowledge-Intensive NLP*. NeurIPS.

Mac Lane, S. (1998). *Categories for the Working Mathematician* (2nd ed.). Springer.

Mitchell, M., Wu, S., Zaldivar, A., et al. (2019). *Model Cards for Model Reporting*. FAT*.

Nye, M., Andreas, J., Bielik, P., et al. (2021). *Show Your Work: Scratchpads for Intermediate Computation*. arXiv:2112.00104.

Olah, C., Cammarata, N., Schubert, L., et al. (2020). *Zoom In: An Introduction to Circuits*. Distill.

Ovadia, Y., Fertig, E., Ren, J., et al. (2019). *Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty under Dataset Shift*. NeurIPS.

Ouyang, L., Wu, J., Jiang, X., et al. (2022). *Training Language Models to Follow Instructions with Human Feedback*. NeurIPS.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2018). *Anchors: High-Precision Model-Agnostic Explanations*. AAAI.

Schölkopf, B., Locatello, F., Bauer, S., et al. (2021). *Toward Causal Representation Learning*. Proceedings of the IEEE, 109(5), 612–634.

Soares, N., Fallenstein, B., Yudkowsky, E., & Armstrong, S. (2015). *Corrigibility*. AAAI Workshop on AI and Ethics.

The Univalent Foundations Program. (2013). *Homotopy Type Theory: Univalent Foundations of Mathematics*. Institute for Advanced Study.

Wang, X., Wei, J., Schuurmans, D., et al. (2022). *Self-Consistency Improves Chain of Thought Reasoning in Language Models*. arXiv:2203.11171.

Wei, J., Wang, X., Schuurmans, D., et al. (2022). *Chain of Thought Prompting Elicits Reasoning in Large Language Models*. NeurIPS.

Yao, S., Yang, J., Zhang, N., et al. (2023). *ReAct: Synergizing Reasoning and Acting in Language Models*. ICLR.

Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2017). *Understanding Deep Learning Requires Rethinking Generalization*. ICLR.

Youvan, D. C. (2025, September). *Deterministic Witnesses, Generative Drafts: The ChatGPT × youvan.ai Hybrid Research Method*. DOI: 10.13140/RG.2.2.24829.22241.

Youvan, D. C. (2025, September). *ai1-Linguo: A CT/HoTT Compliance Bench for Language on a Single PC*. DOI: 10.13140/RG.2.2.31238.64324.

Youvan, D. C. (2025, September). *Achieving Artificial Intelligence (youvan.ai1) on a PC with Category Theory & HoTT*. DOI: 10.13140/RG.2.2.36085.03047.